НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено

Завідувач кафедри

_____ О.В. Коваль
(підпис) (ініціали, прізвище)

“___”_____2019р.

ДИПЛОМНА РОБОТА
на здобуття ступеня бакалавра

з напряму підготовки 6.050101 “ Комп’ютерні науки “

на тему “Візуалізація результатів моделювання робочого процесу газотурбінної установки за допомогою веб-технологій“

Виконав (-ла): студент (-ка) 4 курсу, групи ТР-51

Трофімович Віталій Сергійович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник Доцент, к.т.н, Лабжинський В.А. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший , бакалаврський

Напрям підготовки 6.050101 “ Комп’ютерні науки”

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ О.В. Коваль
(підпис)
”___”_____2019р.

ЗАВДАННЯ
на дипломну роботу студенту
Трофімович Віталій Сергійович
(прізвище, ім’я, по батькові)

1. Тема роботи “ Візуалізація результатів моделювання робочого процесу газотурбінної установки за допомогою веб-технологій “

керівник роботи Лабжинський Володимир Анатолійович, доцент, к.т.н.
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”__”____ 201__р. № __

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи Мова програмування Python у середовищі JetBrains PyCharm 2018 у вигляді веб-сервісу за технологією Flask і jQuery

4.Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити) Ознайомитися з теоретичною частиною моделювання газотурбінної установки, створити систему візуалізації результатів газотурбінної установки. Розробити алгоритм зчитування текстового файлу з результатами роботи газотурбінної установки. Реалізувати систему порівняння робочого процесу ТЕЦ з декількома газотурбінними установками.

5. Перелік ілюстративного матеріалу
«Актуальність розробки», «Мета та завдання роботи», «Моделювання газотурбінної установки», «Засоби розробки», «ККД газотурбінної установки», «Розроблений додаток», «Висновки» .

7. Дата видачі завдання ”__”_____201__р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Вивчення та аналіз задачі	01.12.2018-31.03.2019	
2	Розробка архітектури та загальної структури системи	01-18.04.2019	
3.	Розробка структур окремих підсистем	19-25.04.2019	
4.	Програмна реалізація системи	26.04-13.05.2019	
5.	Оформлення пояснювальної записки	06.05-01.06.2019	
6.	Захист програмного продукту	25.05.2019	
7.	Передзахист	01.06.2019	
8.	Захист	19.06.2019	

Студент _____ Трофімович В.С.
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Лабжинський В.А.
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

Метою дипломної роботи є створення системи візуалізації моделювання робочого процесу газотурбінної установки за допомогою веб-технологій.

Об’єктом дослідження є робочий цикл газотурбінної установки і відображення результатів її роботи. Було виконано огляд існуючих програмних застосунків для створення цієї системи та ознайомлено з усіма характеристиками і циклом роботи газотурбінної установки.

Результатом роботи стало створення системи, яка дозволяє візуалізувати результати моделювання робочого процесу газотурбінної установки.

Загальний обсяг роботи: 57 сторінок, 20 ілюстрацій, 14 бібліографічних посилань та 3 додатки.

Ключові слова: газотурбінна установка, моделювання газотурбінної установки, робочий процес, текстовий файл, система візуалізації, веб-сторінка.

ABSTRACT

The purpose of the thesis is to create a visualization system for simulating the workflow of a gas turbine installation using web technologies.

The object of the study is the working cycle of the gas turbine plant and the reflection of the results of its work. An overview of existing software applications was created to create this system and familiarized with all the features and cycle of the gas turbine installation.

The result of the work was the creation of a system that allows visualization of the results of modeling the workflow of the gas turbine installation.

Total volume of work: 57 pages, 20 illustrations, 14 bibliographic references and 3 appendices.

Key words: gas turbine installation, gas turbine installation modeling, workflow, text file, visualization system, web page.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГТУ — газотурбінна установка

ТЕЦ — теплоелектроцентраль

ККД — коефіцієнт корисної дії

DOM (Document Object Model) – об’єктна модель документа

CDN (Content delivery network) – мережа доправлення і розповсюдження контенту

API (Application Programming Interface) — інтерфейс прикладного програмування

REST (Representational state transfer) — підхід до архітектури мережевих протоколів

JSON (JavaScript Object Notation) — формат даних

HTTP (HyperText Transfer Protocol) —протокол передачі гіпертексту

DI (Dependency Injection) — шаблон для передачі зовнішніх залежностей окремому програмному компоненту (одна з форм інверсії управління)

IDLE (Integrated Development and Learning Environment) – інтегроване середовище розробки і навчання на мові Python

IDE (Integrated Development Environment) – інтегроване середовище розробки

ВСТУП

Актуальність теми. На сьогоднішній день досить широким стало використання різноманітних газотурбінних установок. Принцип роботи такої установки полягає в тому, що стиснене атмосферне повітря з компресора надходить у камеру згоряння, яка також живить паливо, яке, згораючи, утворює велику кількість газоподібних продуктів горіння під високим тиском. Тоді в газовій турбіні енергія тиску продуктів згоряння перетвориться на механічну роботу за рахунок обертання лопаток, частина якої витрачається на стиснення повітря в компресорі. Решта роботи передається на приводний пристрій. Роботу, що споживається цим агрегатом, вважається корисною роботою двигуна. Зважаючи на те, що робочий процес газотурбінної установки вимагає певних затрат ресурсів, а особливо часу, виникає потреба в створенні системи візуалізації результатів моделювання робочого процесу за допомогою веб технологій.

Створена система дозволила би візуалізувати результати роботи установки, та створити порівняльні діаграми її робочого процесу при відповідних вхідних параметрах, які ми отримуємо з текстового файлу. Таким чином, отримуючи певні вхідні параметри, які надходить до користувача з текстового файлу газотурбінної установки, можна здійснювати візуалізацію результатів робочого процесу газотурбінної установки за допомогою сучасних веб-технологій.

В роботі проаналізовано і відображено всі основні параметри які мають свій вплив на хід роботи газотурбінної установки створено діаграми і надана можливість користувачу обрати певний текстовий файл, який він забажає. На основі порівняльного аналізу програмних бібліотек веб-технологій були обрані бібліотеки jQuery і Flask. Вони містять досить ефективні засоби для візуалізації інформації та її досить гармонічного представлення користувачу.

Записка складається з 4 розділів:

В першому розділі описується призначення системи візуалізації результатів моделювання робочого процесу газотурбінної установки та задачі, які розв’язуються системою.

В другому розділі описується аналіз предметної області (опис робочого циклу ГТУ і обчислення ККД).

В третьому розділі описуються засоби розробки системи та програмні середовища розробки.

В четвертому розділі описується програмна реалізація системи, архітектура системи та робота користувача з системою.

1. ЗАДАЧІ ДЛЯ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Метою роботи є створення системи візуалізації результатів моделювання робочого процесу газотурбінної установки за допомогою веб-технологій, яка б дозволяла отримувати результати роботи моделі, зчитувати інформацію з текстового файлу, створювати порівняльні діаграми між обраними ТЕЦ та надати користувачу можливість вибору певних результатів, які йому необхідні.

Під час виконання роботи необхідно вирішити наступні задачі:

– створення системи візуалізації результатів робочого процесу газотурбінної установки, яка б дозволяла відобразити за допомогою веб-браузера параметри системи в певний момент часу роботи системи за допомогою наданого текстового файлу;

– створення системи зчитування текстового файлу з результатами роботи газотурбінної установки;

– розробка системи порівняння результатів робочого процесу газотурбінної установки, яка б дозволяла порівняти результати роботи з використанням декількох газотурбінних установок за допомогою діаграм.

1.1 Задача створення системи візуалізації результатів робочого процесу газотурбінної установки

Вхідними даними для створення системи візуалізації результатів робочого процесу газотурбінної установки є текстовий файл, який містить наступні дані:

– тиск на вході в компресор, Па;
– молярний потік повітря, моль/с;
– температура повітря, К;

– тиск повітря, Па;

– назва ТЕЦ;

– ККД газотурбінної установки;

– назва і порядковий номер газотурбінної установки.

Перед описом вихідних даних, які ми отримаємо після того, як газотурбінна установка завершить свій робочий цикл, сформуємо три стани параметрів газу, у якому він перебуває за весь час роботи газотурбінної установки:

– перший стан описує параметри газу перед тим, як він потрапить в камеру згоряння;

– другий стан описує параметри газу після того, як він потрапив в камеру згоряння;

– третій стан описує параметри газу після того, як він потрапив у саму газову турбіну.

До вихідних даних, які ми отримуємо з текстового файлу відносяться:

– зміна масової витрати газу(Кг/с) для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна температури газу(К) для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна тиску газу(Па) для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна масової частки H_2O(%) в газовій суміші для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна масової частки N_2(%) в газовій суміші для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна масової частки O_2(%) в газовій суміші для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна масової частки CH_4(%) в газовій суміші для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки;

– зміна масової частки CO_2(%) в газовій суміші для кожного із описаних раніше трьох станів в залежності від часу роботи газотурбінної установки.

– ККД газотурбінної установки в залежності від часу роботи газотурбінної установки.

1.2 Задача створення системи зчитування текстового файлу з результатами роботи газотурбінної установки

Для вирішення задачі було обрано мову програмування Python і регулярні вирази.

Регулярний вираз – це рядок, що описує або збігається з безліччю рядків, відповідно до набору спеціальних правил синтаксису. Вони використовуються в багатьох текстових редакторах і допоміжних інструментах для відображення, пошуку і зміни тексту на основі заздалегідь визначених шаблонів.

Для того, щоб зчитати текстовий файл з результатами роботи потрібно виконати такі дії:

– вибрати файл в системі;

– описати файлову змінну;

– зв'язати цю змінну з текстовим файлом;

– обрати файл для зчитування;

– зчитати інформацію за допомогою регулярних виразів і алгоритму зчитування символів з остаточною її перевіркою в консолі;

– відобразити інформацію за допомогою браузера.

Вищенаведені етапи визначають загальну схему зчитування та відображення результатів роботи газотурбінної установки.

В поданому алгоритмі на вхід системи подається вікно з можливістю вибору певного текстового файлу. Після цього в програмі описується змінна, яка зв'язується з текстовим файлом. Коли ці кроки зроблені, за допомогою регулярних виразів зчитується обрана інформація використовуваного текстового файлу, яка потім буде відображена у браузері.

Результатом роботи системи є веб-сторінка, яка дозволяє користувачу переглянути результати роботи газотурбінної установки не відкриваючи текстовий файл перед її запуском, а лише обираючи його з переліку всіх файлів відповідної директорії.

1.3 Задача розробки системи порівняння результатів робочого процесу ТЕЦ з декількома газотурбінними установками

Розроблене програмне забезпечення повинно надавати можливість користувачу створювати порівняльні діаграми між обраними параметрами і переглядати ці діаграми разом із самими результатами роботи газотурбінної установки, при цьому користувач повинен мати змогу самостійно визначати і порівняти обрану кількість газотурбінних установок.

Вхідним параметром є текстовий файл з результати роботи газотурбінної установки. Вихідними параметрами є набір діаграм, які на основі цих результатів показують користувачу тенденцію змін того чи іншого параметру серед усіх газотурбінних установок.

2. ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

В даному розділі буде зроблено огляд принципу роботи газотурбінної установки, на основі якого створюватиметься текстовий файл з результатами її роботи а також розглянуто розрахунок коефіцієнта корисної дії.

2.1 Схема роботи газотурбінної установки

Газотурбінна установка (ГТУ) складається з двох основних частин – силової турбіни і генератора, які розташовані в одному корпусі. Потік високотемпературного газу впливає на лопатки турбіни (створює крутний момент). Утилізація тепла за допомогою теплообмінника або відпрацьованого котла збільшить загальну ефективність установки.

ГТУ може працювати як на рідкому, так і на газоподібному паливі. У нормальному режимі роботи – на газі, а в резервному (аварійному) - автоматично перемикається на дизельне паливо. Оптимальним режимом роботи газотурбінної установки є комбіноване виробництво теплової та електричної енергії. ГТУ може працювати як у базовому режимі, так і для покриття пікових навантажень.

Принцип роботи газотурбінної установки є наступним. Компресор засмоктує повітря з атмосфери, стискає його до певного тиску і подає в камеру згоряння. Тут постійно подається рідке або газоподібне паливо. Спалювання палива за такою схемою відбувається безперервно, при постійному тиску, тому такі ГТУ називаються газотурбінними установками безперервного горіння або ГТУ з горінням при постійному тиску. Гарячі гази, що утворюються в камері згоряння в результаті спалювання палива, надходять у турбіну. У турбіні газ розширюється, а його внутрішня енергія перетворюється в механічну. Відпрацьовані гази залишають турбіну через викиди в навколишнє середовище (в атмосферу).

Частина потужності, що розвивається газовою турбіною, витрачається на обертання компресора, а решта (корисна потужність) надається споживачеві.

Потужність, споживана компресором, є відносно великою і в простих схемах при помірній температурі робочого середовища може в 2-3 рази перевищувати корисну потужність ГТУ. Це означає, що повна потужність газової турбіни буде набагато більшою, ніж корисна потужність ГТУ.

Оскільки газова турбіна може працювати тільки при наявності стисненого повітря, що надходить тільки від компресора, який обертається турбіною, очевидно, що початок запуску ГТУ має бути від стороннього джерела енергії (пускового двигуна), через який компресор обертається до тих пір, поки камера згоряння не почне отримувати газ певних параметрів і в кількості, достатній для початку роботи газової турбіни. Як видно з схеми, в якості палива вибирається метан.

Схема роботи газотурбінної установки зображена на рисунку 2.1.

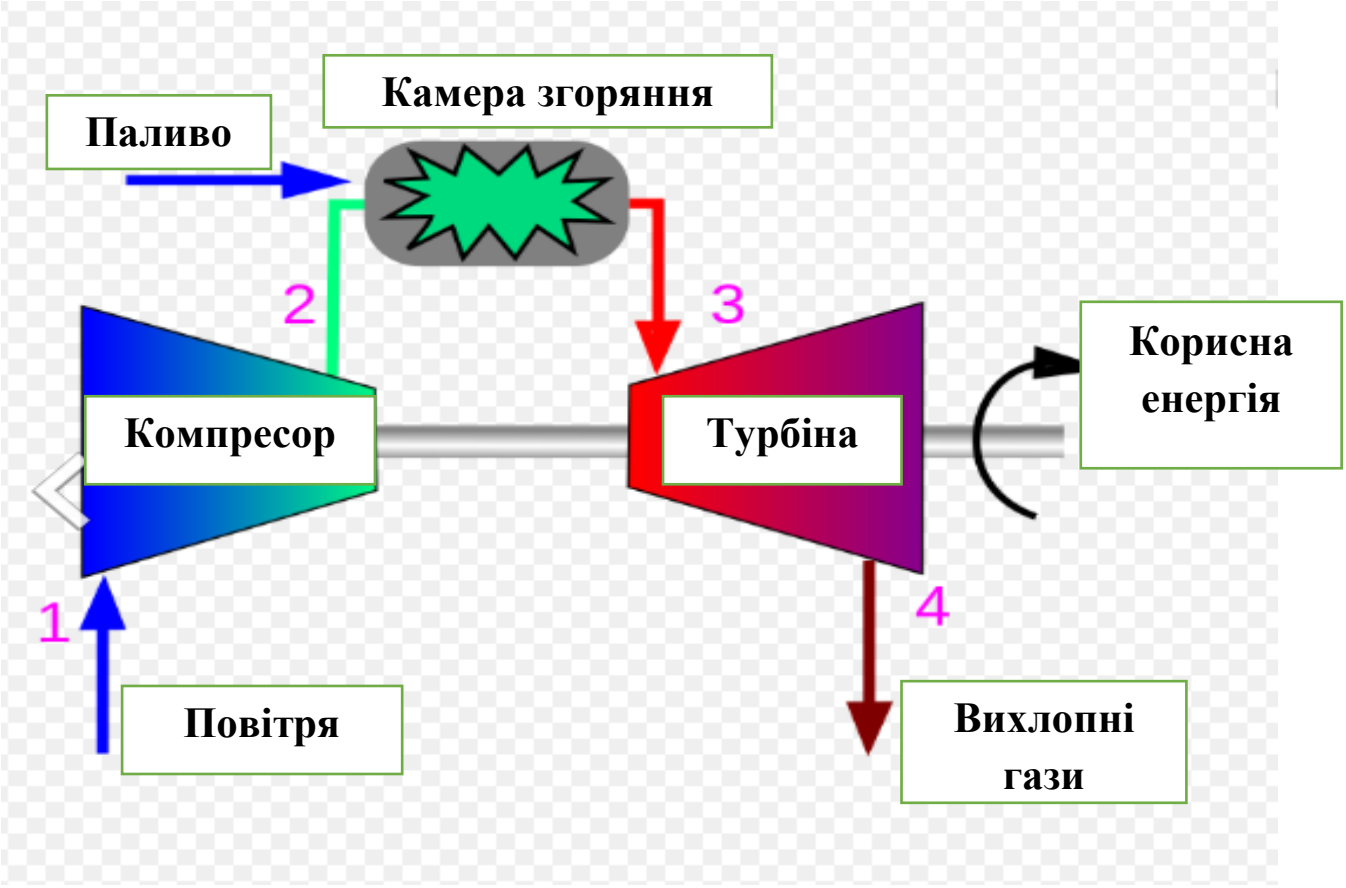

Рисунок 2.1 – Схема роботи газотурбінної установки

Робочий процес газотурбінної установки описується за допомогою циклу Брайтона (Brayton Cycle). Цикл Брайтона складається з процесів:

– ізоентропійне стиснення – зміна стану термодинамічної системи, коли не змінюється її ентропія;

– ізобарне розширення – термодинамічний процес, який відбувається при сталому тиску;

– ізоентропійне розширення – зміна стану термодинамічної системи, коли не змінюється її ентропія;

– ізобарне стиснення – термодинамічний процес, який відбувається при сталому тиску.

2.2 Обчислення коефіцієнта корисної дії газотурбінної установки

ККД газотурбінної установки, для якої в якості палива використовується метан обчислюється за формулою (2.1):

$$\mu = \frac{P_3 - P_1 - P_2}{M * 50010 * 16} * 100\%, \quad (2.1)$$

де μ – ККД установки, %;

P_1 – тиск на виході з компресора повітря, Па;

P_2 – тиск на виході з компресора метану, Па;

P_3 – тиск на виході з турбіни, Па;

M – молярний потік метану, моль/с.

Методи підвищення ККД ГТУ:

1) Використання теплоти викидаючих газів(підігрів стисненого повітря продуктами згорання газової турбіни);

2) Проміжне охолодження повітря в компресорі;

3) Проміжний підігрів газів в газовій турбіні.

3. ЗАСОБИ РЕАЛІЗАЦІЇ СИСТЕМИ

Веб-технології – галузь з величезним вибором інструментів, бібліотек і фреймворків. Однак велика кількість варіантів призводить до плутанини і розуміння. З великим простором для творчості та експериментів важливо не розгубитись при виборі правильного інструменту.

Вибір структури в майбутньому може бути або перевагою, або недоліком майбутнього проекту. Фреймоворки диктують правила побудови архітектури програмного забезпечення. Навіть на початковій стадії розробки визначається поведінка за замовчуванням, яка повинна бути розширена та змінена відповідно до вимог у процесі розробки, а зміна архітектури проекту є дуже дорогою процедурою. З іншого боку, використання конкретного фреймворку дозволяє легко підтримувати, модернізувати і масштабувати проект, порівняно легко реалізувати будь-які бізнес-процеси. Рамкові рішення, як правило, працюють набагато швидше, безпечніше і більш чутливо, ніж системи самозапису.

Отже, вибір засобів розробки є важливим етапом у розробці програмного забезпечення, і цей розділ буде охоплювати основні підходи та програмні засоби, їх переваги та недоліки для розробки системи візуалізації результатів моделювання робочого процесу газотурбінної установки.

3.1 Аспекти при виборі бібліотек

Існує декілька факторів, які істотно впливають на вибір бібліотек. Всі вони необхідні по-своєму і ступінь важливості кожного з них залежить від завдання задачі. Деякі бібліотеки є лідерами в певних областях, але вони погано взаємодіють з іншими, і є такі, які є комбінацією всіх факторів на задовільному рівні.

Давайте розглянемо основні аспекти при виборі бібліотек.

– наявність навчальних матеріалів. Важливий, але зазвичай недооцінений людьми аспект. Деякі бібліотеки можуть бути представлені в простих і

елементарних прикладах, але в процесі розробки можуть виникнути ситуації, коли необхідно додатково вивчити можливості інструмента і, крім сухої документації, знадобляться додаткові ресурси, наприклад, курси, книги і статті. Якісні навчальні ресурси значно скоротять і зроблять процес навчання цікавим для людини;

– популярність. Вивчення екзотичних технологій може бути цікавим процесом, але на практиці краще віддавати перевагу "випробуваному часом" інструменту. Для цього є кілька причин. Якщо бібліотеки не дуже популярні, то це означає, що в них немає так багато розробників та фахівців які досконало володіють цими технологіями. Іншою причиною є відсутність людей, з якими можна проконсультуватися і залишитися наодинці з документацією;

– основні можливості та можливості інструмента. Цей фактор безпосередньо вказує на те, як і що можна зробити з обраним інструментом, які обмеження він буде накладати на програмне забезпечення і які переваги та недоліки будуть у користувача при взаємодії з цим інструментом;

– простота використання. Найкращий спосіб зрозуміти, як працює бібліотека – це використовувати його у простому проекті. Це дасть змогу зрозуміти, чи дозволяє він розв'язати проблеми, перелічені вище. При роботі з бібліотеками слід також враховувати продуктивність розвитку, а також рівень залежності бібліотеки від інших інструментів. Варто звернути увагу на доступні програмні засоби, такі як IDE і розширення;

– швидка інтеграція з іншими послугами. Цей аспект тісно пов'язаний з попереднім. Не має значення, наскільки обрана структура функціональна, все ще є шанс зіткнутися з проблемою, для вирішення якої знадобляться додаткові інструменти. Існує багато бібліотек, які зосереджуються на вирішенні індивідуальних завдань. Якщо ви намагаєтеся підключити щось кожен раз, коли це займає кілька годин - це, мабуть, не найкращий варіант того, що могло би відбутись, якщо б ви вибрали правильну технологію.

При розробці програмного продукту важливим чинником є правильний вибір технологій та засобів програмної реалізації. Основним середовищем розробки було JetBrains PyCharm 2018.2.

Для створення інтерфейсу використовувалися JavaScript і jQuery.

Для розробки алгоритмів використовувалась мова Python і бібліотека Flask.

В якості інструменту для побудови графіків було використано Google Charts.

3.2 Мова програмування Python

Python є інтерпретованою мовою програмування високого рівня загального призначення. Створена Гвідо ван Россумом і вперше випущена в 1991 році, філософія дизайну Python підкреслює читабельність коду з його помітним використанням значних пробілів. Його мовні конструкції та об'єктно-орієнтований підхід спрямовані на те, щоб допомогти програмістам написати чіткий, логічний код для малих і великих проектів.

В Python вбудований динамічний збиральник сміття. У галузі комп'ютерних наук збір сміття є формою автоматичного управління пам'яттю. Він підтримує багато парадигм програмування, включаючи процедурні, об'єктно-орієнтовані та функціональні програми. Python часто описується як "підключена бібліотечна" мова через його всеосяжну стандартну бібліотеку.

Python був задуманий в кінці 1980-х років як наступник мови ABC. Python 2.0, що був випущений у 2000 році, представив такі функції, як розуміння списків і система автоматичного управління пам'яттю, здатна збирати еталонні цикли. Python 3.0, випущений у 2008 році, був серйозною ревізією мови, яка не є повністю зворотно сумісною, і багато Python 2-проектів не працює без змін на Python 3. Через занепокоєння щодо кількості коду, написаного для Python 2, підтримка Python 2.7 (останній реліз був в серії 2.x) був розширений до 2020 року. Розробник мови Гвідо ван Россум взяв на себе відповідальність за проект до липня 2018 року, але тепер поділяє його керівництво як член ради управління з п'яти осіб.

Інтерпретатори Python доступні для багатьох операційних систем. Світове співтовариство програмістів розробляє та підтримує CPython- інтерпретатор байт-

коду, написаний на C. Некомерційна організація Python Software Foundation направляє ресурси для розвитку Python та CPython. Крім CPython, існують інші реалізації Python: Jython, IronPython, PyPy, Stackless Python.

Python – це мова програмування з декількома парадигмами. Об'єктно-орієнтоване програмування і структуроване програмування повністю підтримуються, і багато з його функцій підтримують функціональне програмування і аспектно-орієнтоване програмування (у тому числі метапрограмування і метаоб'єкти (магічні методи)). Багато інших парадигм підтримуються за допомогою розширень, включаючи в себе розробку і логічне програмування.

Python використовує динамічну типізацію, а також комбінацію підрахунку опорних даних і колектора виявлення сміття для керування пам'яттю. Вона також має динамічну роздільну здатність імен (пізнє зв'язування), яка пов'язує метод і імена змінних під час виконання програми.

Дизайн Python пропонує певну підтримку функціонального програмування в традиції Lisp. Він має функції фільтрації, карти та зменшення. Також тут присутні перелік списків, словники, множини та генераторні вирази. Стандартна бібліотека має два модулі (itertools і functools), які реалізують функціональні інструменти, запозичені з Haskell і Standard ML.

Серед основних її переваг можна назвати такі:

– чистий синтаксис (необхідно використовувати відступи для вибору блоків);

– портативність програм (властива більшості інтерпретованих мов);

– стандартний розподіл має велику кількість корисних модулів (включаючи модуль розробки графічного інтерфейсу);

– можливість використання Python в діалоговому режимі (дуже корисно для експериментів і вирішення простих завдань);

– стандартний дистрибутив має просте, але потужне середовище розробки з ім'ям IDLE, написане на Python;

– зручний для вирішення математичних задач (має засоби для роботи з комплексними числами, може працювати з цілими числами довільного значення, в режимі діалогового вікна можна використовувати як потужний калькулятор);

– з відкритим вихідним кодом (здатність редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня і простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів і той факт, що він є інтерпретованою мовою, робить його ідеальним для створення сценаріїв і швидкого розвитку програм у багатьох галузях на більшості платформ.

На рисунку 3.1 докладно зображено архітектуру мови програмування Python.

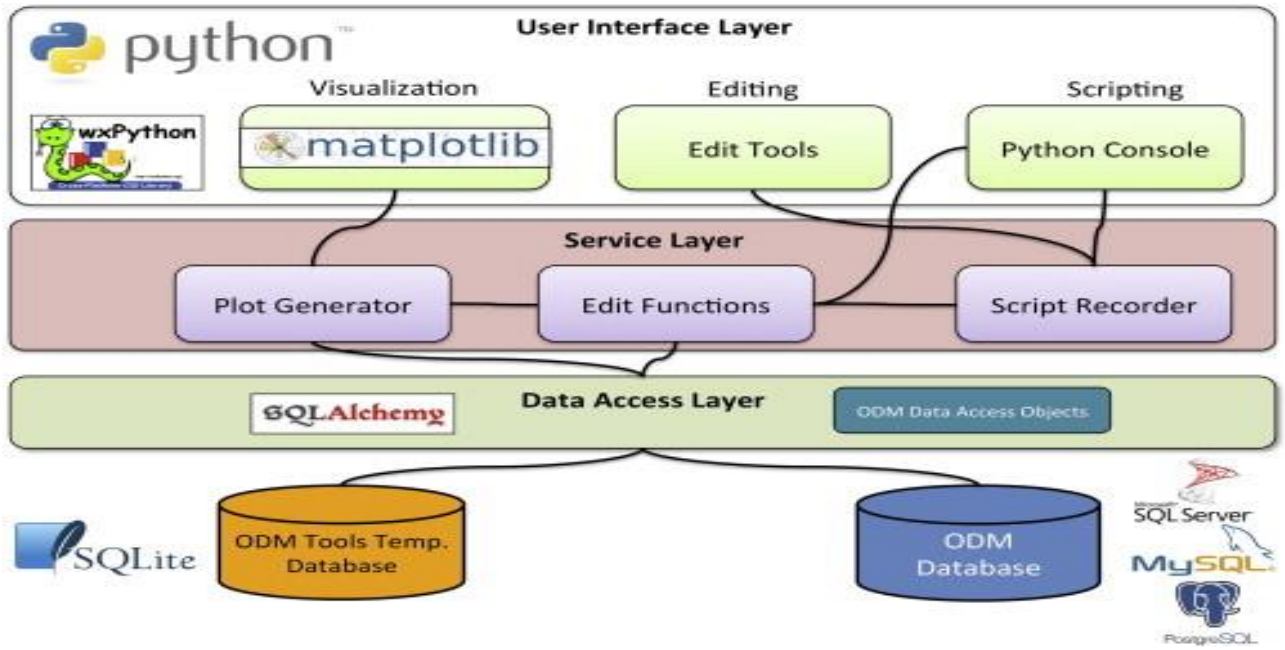

Рисунок 3.1 – Архітектура Python

Існує ще один аспект цієї інформації. Наведені вище тези не передають новий синтаксис, удосконалення основних класів, доступних виключно в Python 3, які дозволяють використовувати нові шаблони і роблять суттєвий вплив на швидкість і продуктивність розвитку. Іншими словами, Python 3 є більш потужною і виразною мовою.

3.3Бібліотека Flask

Flask – це бібліотека, написана на Python. Вона класифікується як мікробібліотека, оскільки не вимагає спеціальних інструментів або бібліотек. Мікробібліотека, як правило, не має ніяких залежностей із зовнішніми бібліотеками. Це має свої плюси і мінуси. Плюси полягають у тому, що фреймворк є доволі легким, існує невелика залежність від оновлення і спостереження за помилками безпеки, але це означає, що деякий час користувачам доведеться більше працювати самостійно або збільшувати список залежностей, додаючи плагіни для роботи над додатковими функціями. Він не має рівня абстракції бази даних, перевірки форм або будь-яких інших компонентів, де вже існують бібліотеки третіх сторін, які надають загальні функції. Однак Flask підтримує розширення, які можуть додавати функції програми, якщо б вони були реалізовані в самому Flask. Багаточисельні розширення забезпечують інтеграцію баз даних. Існують розширення для об'єктно-реляційних мапперів, валідації форм, обробки завантажень, різних відкритих технологій аутентифікації та декількох спільних інструментів, пов'язаних з формами. Розширення оновлюються більш регулярно, ніж основна програма Flask. Flask є мікробібліотекою, але він готовий до використання в дуже складних і цікавих цілях. На рисунку 3.2 докладно зображено підключення бібліотеки в середовищі програмування PyCharm.

Flask пропонує широкий спектр можливостей, але при цьому не застосовує ніяких залежностей або макетів від інших бібліотек проекту. Розробник самостійно обирає інструменти та бібліотеки, які він хоче використовувати. Бібліотека надає багато розширень, що полегшує додавання нових функцій. Програми, які використовують фреймворк Flask, включають в себе Pinterest, LinkedIn, і веб-сторінку спільноти для самої Flask.

```python
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return "Hello, Flask!"

if __name__ == '__main__':
    app.run(debug=True)
```

Рисунок 3.2 – Підключення Flask

Flask був створений Арміном Ронахером з Pocoo та міжнародною групою ентузіастів Python, яка була сформована в 2004 році. На думку Роначера, ідея була спочатку жартом серед колег, який був досить популярний, щоб зробити його серйозним застосуванням.

Коли Ronacher і Georg Brandl створили систему дошки оголошень, написану на Python, були розроблені проекти Pocoo Werkzeug і Jinja.

Незважаючи на відсутність великого попиту, Flask став популярним серед ентузіастів Python. З середини 2016 року це була найпопулярніша структура веб-розробки Python на GitHub.

Мікрофреймворк Flask базується на проектах Pocoo Werkzeug і Jinja2.

Werkzeug - це бібліотека утиліти для мови програмування Python, тобто інструментарій для додатків інтерфейсу веб-сервера і ліцензований під ліцензією BSD. Werkzeug може реалізувати об'єкти програмного забезпечення для запитів, відповідей і функцій утиліти. Він може бути використаний для створення користувацького програмного забезпечення поверх нього і підтримує Python 2.6, 2.7 і 3.3.

Jinja2, також Ronacher, є шаблоном для мови програмування Python і ліцензований під ліцензією BSD. Як і у веб-фреймворку Django, він передбачає, що шаблони оцінюються в пісочниці.

1.1 Бібліотека jQuery

jQuery – це бібліотека JavaScript, розроблена для спрощення обходу і маніпулювання деревом HTML DOM, а також для обробки подій, анімації CSS і Ajax. Це безкоштовне програмне забезпечення з відкритим вихідним кодом, що використовує ліцензію MIT. Станом на травень 2019 року jQuery використовується в 73% з 10 мільйонів найпопулярніших веб-сайтів. Веб-аналіз свідчить про те, що він є найбільш широко розгорнутою бібліотекою JavaScript з великим відривом, що має від 3 до 4 разів більше користувачів, ніж будь-яка інша бібліотека JavaScript.

Синтаксис jQuery розроблений для полегшення навігації по документу, вибору елементів DOM, можливості створення анімації, обробки подій та розробки додатків Ajax. jQuery також надає розробникам можливість створювати плагіни поверх бібліотеки JavaScript. Це дає змогу розробникам створювати абстракції для низькорівневої взаємодії та анімації, поліпшення ефектів і високорівневих тематичних віджетів. Модульний підхід до бібліотеки jQuery дозволяє створювати потужні динамічні веб-сторінки та веб-програми.

Набір основних функцій jQuery - вибір елементів DOM, обхід і маніпуляція, що були включені за допомогою свого механізму селектора (названий "Sizzle" з версії 1.3), створив новий "стиль програмування", які зараз ми називаємо алгоритмами і структурами даних DOM. Цей стиль вплинув на архітектуру інших фреймворків JavaScript, таких як YUI v3 і Dojo, що пізніше стимулювало створення стандартного API Selectors.

Microsoft і Nokia використовують jQuery на своїх платформах. Корпорація Майкрософт включила її у Visual Studio для використання в рамках ASP.NET AJAX і ASP.NET MVC Microsoft, в той час як Nokia інтегрувала його в платформу розробки віджету Web Run-Time. jQuery має такий основний набір функцій:

– маніпулювання DOM на основі CSS-селекторів, які використовують імена та атрибути елементів, такі як id та class, як критерії для вибору вузлів у DOM;

– події;

– ефекти та анімації;

– Ajax;

– відкладені та діючі об'єкти для керування асинхронною обробкою;

– синтаксичний аналіз JSON;

– розширюваність за допомогою плагінів;

– утиліти, такі як виявлення функцій.

– способи сумісності, які доступні в сучасних браузерах, але потребують зворотного зв'язку для старих браузерів, таких як jQuery.inArray () і jQuery.each ().;

Бібліотека jQuery зазвичай розподіляється як один файл JavaScript, який визначає всі його інтерфейси, включаючи функції DOM, події та Ajax. Вона може бути включена у веб-сторінку за допомогою посилання на локальну копію або шляхом посилання на одну з багатьох копій, доступних з публічних серверів. jQuery має мережу доставки контенту (CDN), що розміщується на сервісі MaxCDN.
На рисунку 3.4 докладно зображено архітектуру бібліотеки jQuery.

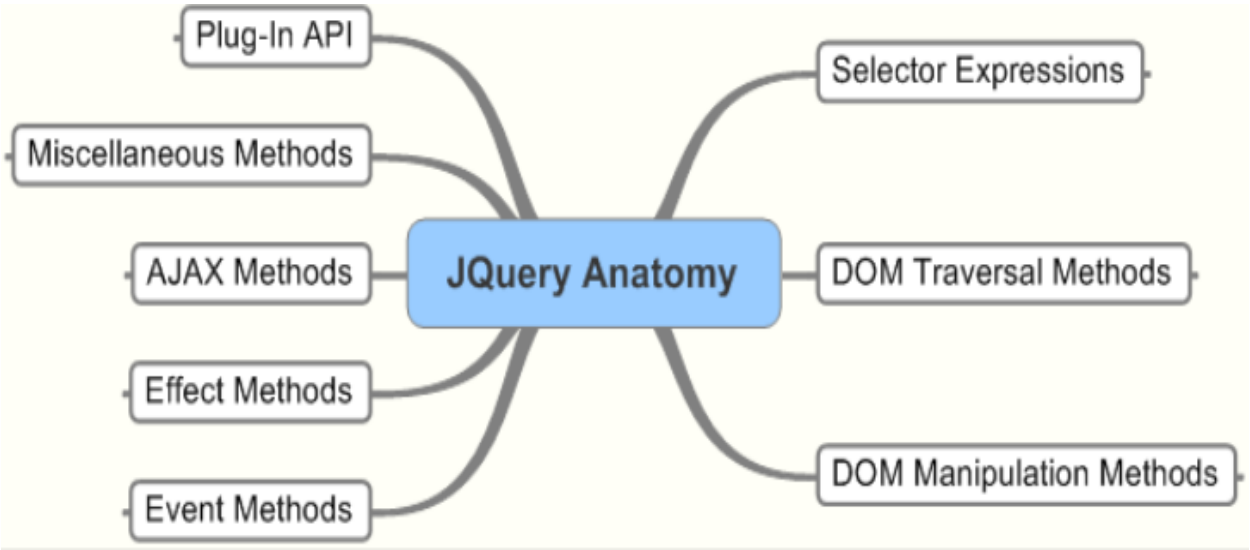

Рисунок 3.4 – Архітектура jQuery

Архітектура запитів дозволяє розробникам створювати плагін для розширення своєї функції. В інтернеті доступні тисячі плагінів jQuery, які охоплюють цілий ряд функцій, таких як Ajax-помічники, веб-служби, датчики даних, динамічні списки, інструменти XML і XSLT, перетягування, події, керування cookie та модальні вікна.

Важливим джерелом плагінів jQuery є домен плагінів на веб-сайті проекту jQuery. Плагіни в цьому домені зараз є відновлені, однак, були випадково видалені в грудні 2011 року, щоб спробувати звільнити сайт від спаму. Новий сайт є сховищем GitHub, що вимагає від розробників повторної подачі плагінів і відповідності новим вимогам до подання. jQuery надає "Центр навчання", який може допомогти користувачам зрозуміти JavaScript і розпочати розробку плагінів jQuery.

3.5 Мова програмування JavaScript

JavaScript, часто скорочено як JS, є мовою програмування високого рівня, що відповідає специфікації ECMAScript. JavaScript має синтаксис фігурних дужок, динамічну типізацію, об'єктно-орієнтаційну орієнтацію на основі прототипу та функції першого класу.

Поряд з HTML і CSS, JavaScript є однією з основних технологій World Wide Web. JavaScript дозволяє створювати інтерактивні веб-сторінки і є невід'ємною частиною веб-додатків. Переважна більшість веб-сайтів використовують її, а основні веб-браузери мають спеціальний JavaScript-механізм для її виконання.

Як багатофункціональна мова, JavaScript підтримує функціональні та імперативні (включаючи об'єктно-орієнтовані та прототипні) стилі програмування. Він має API для роботи з текстом, масивами, датами, регулярними виразами та DOM, але сама мова не включає жодного вводу-виводу, такого як мережа, сховище або графічні засоби. Вона заснована на хост-середовищі, в якій вона вбудована, щоб забезпечити ці функції.

Спочатку реалізовані лише клієнтські веб-браузери, двигуни які містять в собі JavaScript зараз вбудовано в багато інших типів програмного забезпечення, включаючи серверні веб-сервери та бази даних, а також такі, як текстові процесори та програмне забезпечення PDF, а також у середовищі виконання, що робить JavaScript доступним для написання мобільних і настільних додатків, включаючи віджети робочого столу.

На рисунку 3.5 докладно зображено архітектуру мови програмування JavaScript.

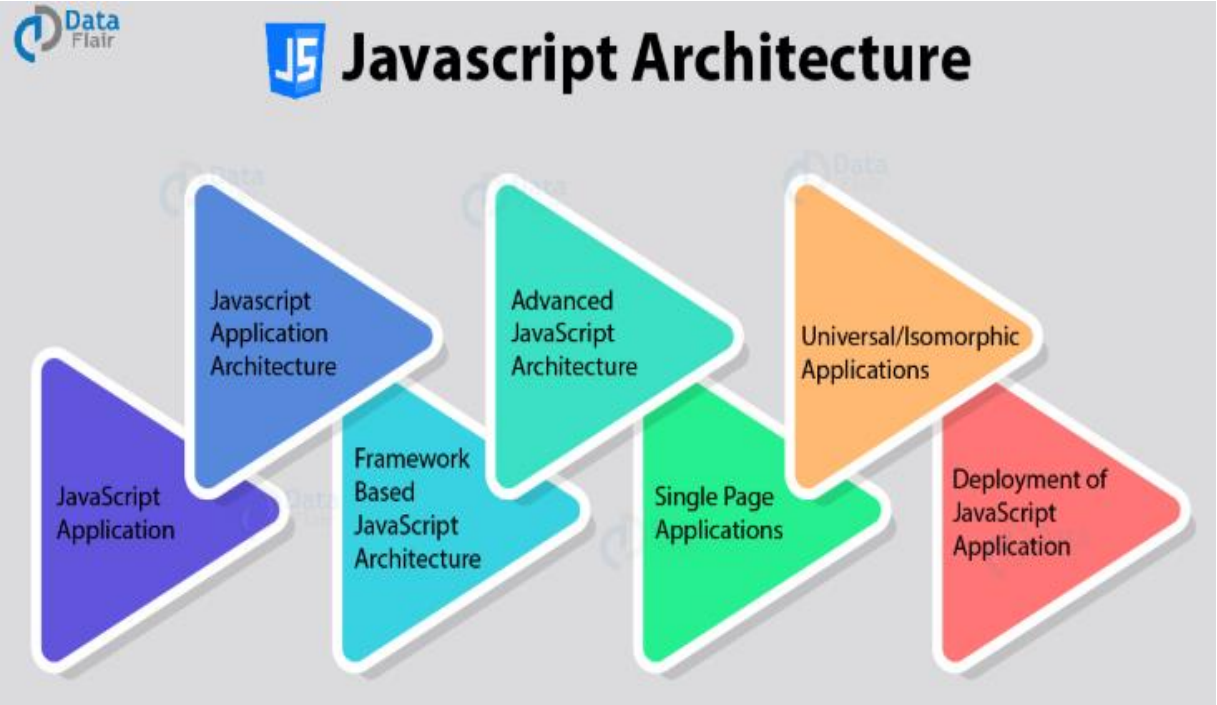

Рисунок 3.5 – Архітектура JavaScript

Терміни Vanilla JavaScript і Vanilla JS посилаються на JavaScript, не розширений жодними фреймворками або додатковими бібліотеками. Сценарії, написані на Vanilla JS, є звичайним JavaScript кодом.

Незважаючи на наявність подібності між JavaScript і Java, включаючи назву мови, синтаксис і відповідні стандартні бібліотеки, ці дві мови є різними і широко варіюються в дизайні. На JavaScript впливали такі мови програмування, як Self і Scheme. Формат серіалізації JSON, який використовується для зберігання структур даних у файлах або передачі їх по мережах, заснований на JavaScript.

Основними можливостями JavaScript є наступне:

– додавання різні ефектів анімації;

– реагування на події - обробляти переміщення покажчика миші, натискання клавіш з клавіатури;

– здійснювати перевірку введення даних в поля форми до відправки на сервер, що в свою чергу знімає додаткове навантаження з сервера;

– створення і зчитування cookie, витягування даних про комп'ютер користувача визначається браузером;

– зміна вмісту HTML-елементів, додавання нових тегів, зміна стилів.

3.6 Шаблон Google Charts

Google Charts – це інтерактивна веб-служба, яка створює графічні діаграми з наданої користувачем інформації. Користувач надає дані та специфікацію форматування, виражену в JavaScript, який вбудовано в веб-сторінку; у відповідь служба посилає зображення діаграми.

Функціональні можливості Google Charts Tools включають в себе:

– динамічні піктограми;

– карти;

– циферблати і дисплеї;

– формули;

– QR-коди;

– можливість створювати свої інструменти візуалізації і використовувати сторонні ресурси.

Графіки Google надають ідеальний спосіб візуалізації даних на своєму веб-сайті. Від простих лінійних діаграм до складних ієрархічних карт дерев, галерея діаграм надає велику кількість готових до використання типів діаграм.

Найбільш поширений спосіб використання графіків Google це простий JavaScript, який користувач вбудовує в свою веб-сторінку. Користувач завантажує деякі бібліотеки діаграм Google, перелічує дані, які потрібно накреслити, вибирає параметри для налаштування діаграми і, нарешті, створює об'єкт діаграми з вибраним ідентифікатором.

Діаграми відображаються як класи JavaScript, а графіки Google надають багато типів діаграм. Користувач завжди може налаштувати діаграму відповідно до

зовнішнього вигляду сайту. Діаграми дуже інтерактивні та розкривають події, які дозволяють підключати їх, створюючи складні панелі інструментів або інші функції, які можуть бути потім інтегровані з веб-сторінкою. Графіки відображаються за допомогою технології HTML5 / SVG для забезпечення сумісності між браузерами (включаючи VML для старих версій IE) і перенесення крос-платформових можливостей для iPhone, iPad та Android. Користувачам ніколи не доведеться зіштовхуватися з плагінами або будь-яким програмним забезпеченням, якщо вони цього не схочуть. Якщо у них є веб-браузер, вони можуть бачити всі графіки і всі можливості, які надає Google Charts.

Доступність Google Charts на інших платформах, відмінних від Windows, робить його найкращим кандидатом для використання всіма розробниками, включаючи розробників Mac і Linux, а також дає всієї платформі можливість використовувати зворотний зв'язок у системі. Цей зворотний зв'язок призводить до створення продукту, який працює краще для всіх користувачів і дозволяє Google просувати свої додатки дуже швидко вперед та займати одні з найвищих позицій на ринку програмних продуктів.

Всі типи діаграм заповнюються даними за допомогою класу DataTable, що полегшує перемикання між типами діаграм під час експерименту, щоб знайти ідеальний вигляд. DataTable надає методи для сортування, модифікації та фільтрування даних, і може бути заповнена безпосередньо з веб-сторінки, бази даних або будь-якого постачальника даних, що підтримує протокол джерел даних для інструментів діаграм.

На рисунку 3.6 докладно зображено можливості Google Charts.

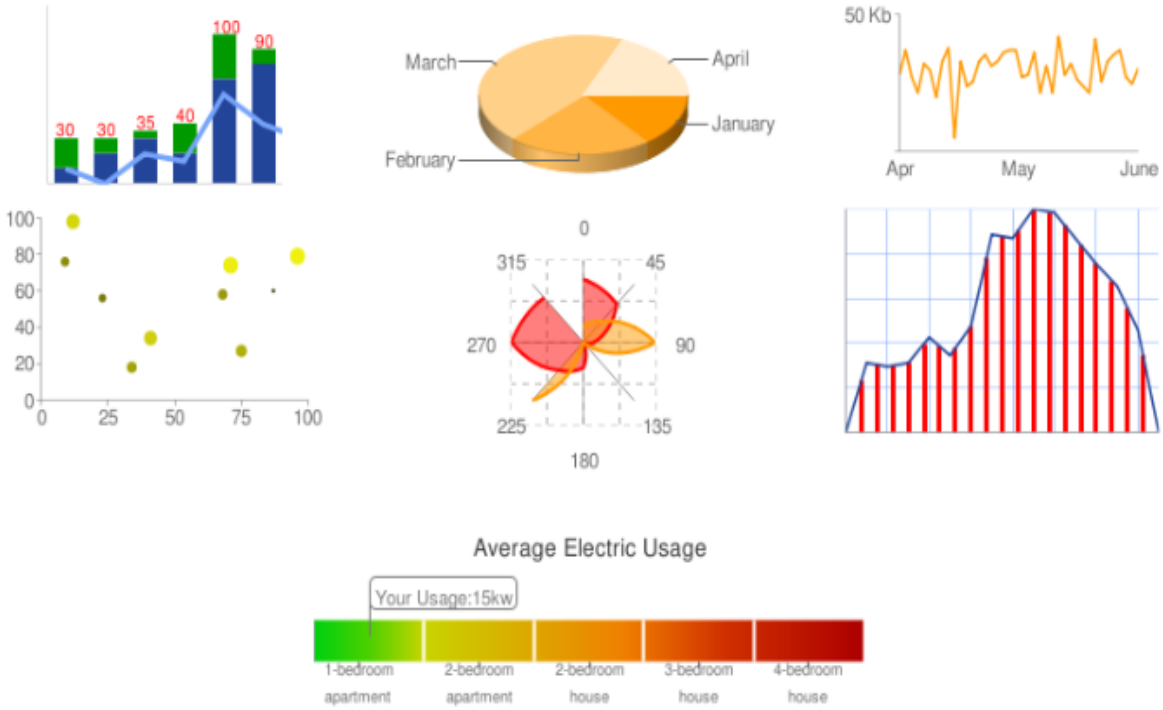

Рисунок 3.6 – Можливості Google Charts

Цей протокол включає SQL-подібне мове запитів і реалізований за допомогою Google Spreadsheets, Google Fusion Tables і постачальників даних третіх сторін, таких як SalesForce. Користувач може реалізувати протокол на власному веб-сайті і стати постачальником даних для інших служб.)

3.7 Середовище JetBrains PyCharm

PyCharm є інтегрованим середовищем розробки (IDE), що використовується в комп'ютерному програмуванні, зокрема для мови Python. Він розроблений чеською компанією JetBrains. Він надає аналіз коду, графічний відладчик, інтегрований тестер модулів, інтеграцію з системами контролю версій (VCSes) і підтримує веб-розробку з Django, а також Data Science з Anaconda.

На рисунку 3.7 докладно зображено інтерфейс середовища PyCharm.

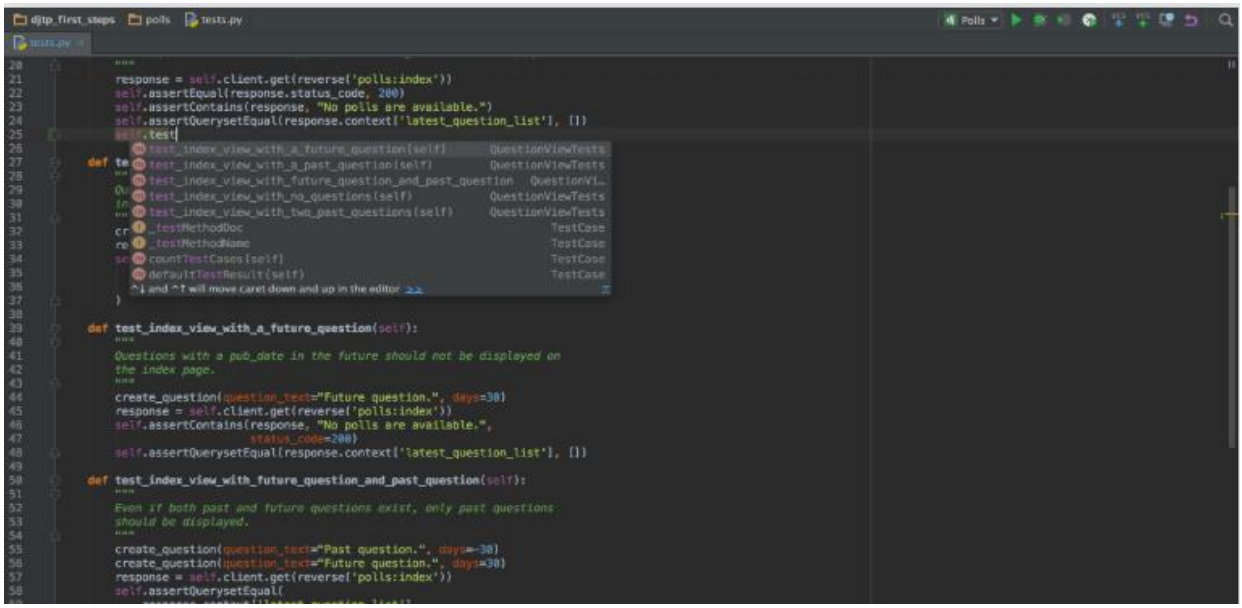

Рисунок 3.7 – Інтерфейс середовища PyCharm

PyCharm є крос-платформним, з версіями Windows, MacOS і Linux. Community Edition випускається під ліцензією Apache, а також є Professional Edition з додатковими функціями, які випущено під власною ліцензією.

Основні можливості PyCharm:

– допомога та аналіз кодування, з завершенням коду, підсвічуванням синтаксису та помилок, інтеграція з linter та швидкими виправленнями;

– проектування та кодова навігація: спеціалізовані види проектів, перегляд структури файлів та швидке перемикання між файлами, класами, методами та застосуваннями;

– рефакторинг Python: включаючи перейменування, метод екстракції, введення змінної, введення константи, витягування, натискання та інші;

– підтримка веб-фреймворків: Django, web2py і Flask;

– інтегрований відладчик Python;

– швидкий перегляд документації для будь-якого елемента прямо у вікні редактора, перегляд зовнішньої документації через браузер, підтримка docstring, генерація, підсвічування, автодоповнення і багато іншого.

4.ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ СИСТЕМИ ВІЗУАЛІЗАЦІЇ РЕЗУЛЬТАТІВ МОДЕЛЮВАННЯ РОБОЧОГО ПРОЦЕСУ ЗА ДОПОМОГОЮ ВЕБ-ТЕХНОЛОГІЙ

Розроблений програмний комплекс створено з використанням веб-технологій і тому працює в браузерах, які підтримують актуальні на сьогоднішній день веб-стандарти. Клієнтська частина використовує компоненти Python для відображення результатів в текстовому файлі, тому існує прив’язка даних між текстовим файлом та регулярними виразами. Таким чином, при зміні або виборі іншого текстового файлу, відповідна частина шаблону змінюється автоматично, тому дані завжди є актуальними та не потребують постійного оновлення сторінки після виконання будь-яких змін. Також реалізовано систему вибору текстового файлу для зручності роботи з системою. У розділі розглядаються такі питання, як візуалізація результатів роботи, зчитування текстового файлу і його вибір та створення порівняльних діаграм за певними параметрами.

4.1 Інсталяція та системні вимоги

Система складається з двох частин. Для використання клієнтської частини користувачу потрібний лише браузер та стабільне інтернет-з’єднання для підтягування даних з Bootstrap і jQuery.

Для надання можливості роботи з системою необхідно розмістити серверну та клієнтську частину систему на сервері. Дана система є доступною для подальшого використання в більш складних проектах.

4.2 Інструкція з використання програмного продукту

Для початку користування системою, користувачу необхідно запустити проект Flask за допомогою інтерпретатора Python, в моєму випадку це JetBrains PyCharm і після компілювання перейти за посиланням http://127.0.0.1:5000/start_page просто натиснувши на це посилання (це показано на рисунку 4.1). Після цих дій відкриється ваш браузер за замовчуванням і сама веб-сторінка за цією адресою. Також користувач може просто скопіювати посилання після компіляції і відкрити її в будь-якому браузері, який інстальовано на його комп'ютері.

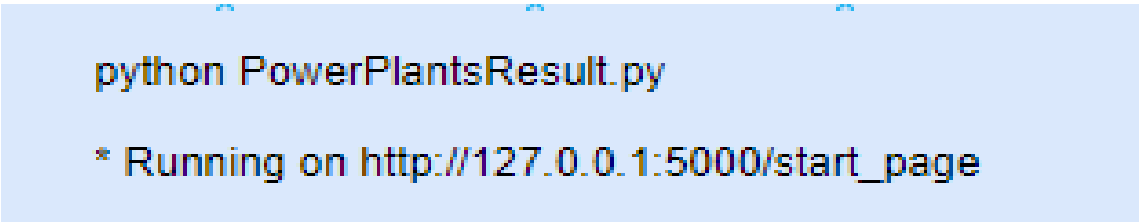

Рисунок 4.1 — Запуск програми

Якщо ви запустите сервер, ви помітите, що він доступний тільки з вашого власного комп'ютера, а не з будь-якого іншого в мережі. Так зроблено за замовчуванням, тому що в режимі налагодження системи користувач програми може виконати код на Python на вашому комп'ютері.

Якщо у вас відключена опція debug або ви довіряєте користувачам в мережі, ви можете зробити сервер публічно доступним, просто змінивши виклик методу run () таким ось чином(рисунок 4.2).

```
app.run(host='0.0.0.0')
```

Рисунок 4.2 — Зміна методу Run

Це вкаже вашій операційній системі, щоб вона слухала мережу з усіх публічних IP-адрес, і користувач зможе запускати сервер на будь-якій машині, яка має вільний доступ до мережі інтернет. Результатом виконання запиту на сервері є токен, який є унікальним для користувача і дозволяє за ним на сервері однозначно визначити користувача і машину, з якої було відправлено запит. Жетон зберігається на клієнтській частині та використовується для доступу до системи,

використовуючи HTTP-атрибут “Open”. При успішно зроблених кроках, які були описані раніше, здійснюється перехід на компонент, відповідальний за відображення вибору файлів з усіх директорій комп'ютера(рисунок 4.3).

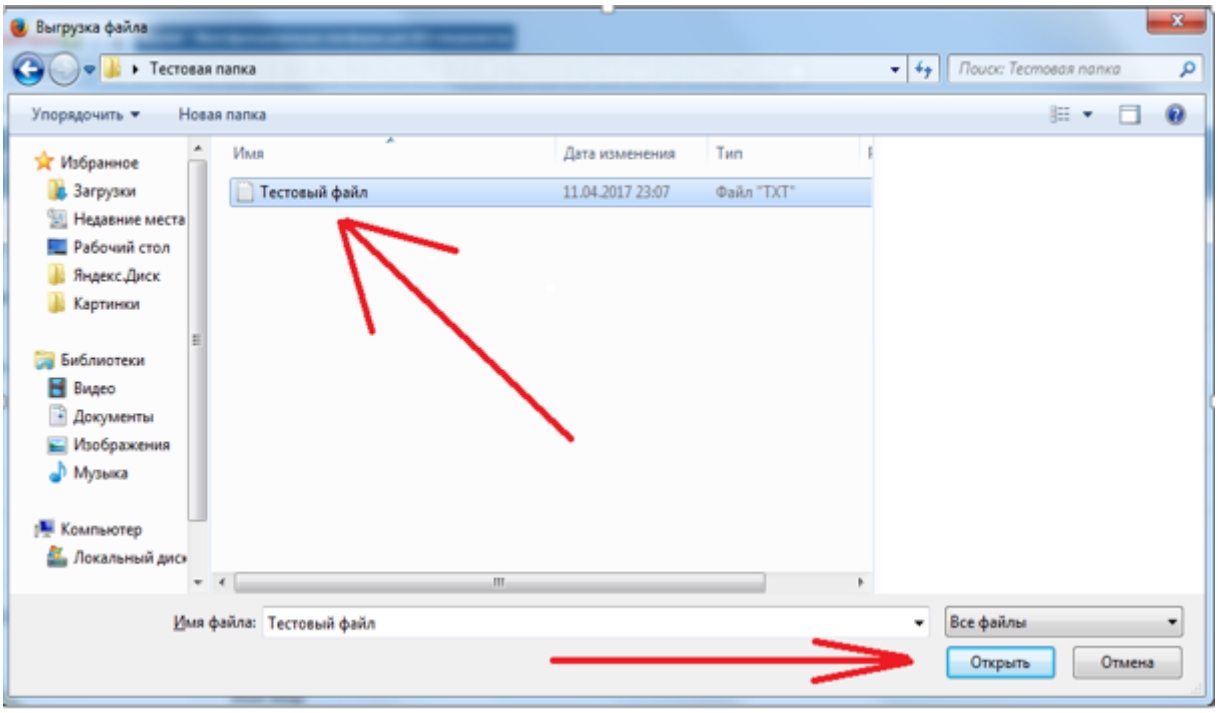

Рисунок 4.3 — Вибір текстового файлу

Після цього користувач повинен обрати текстовий файл, з результатами якого він хоче ознайомитись. Є можливість додавати, видаляти та змінювати параметри результатів, які будуть відображатися, але для цього необхідно спочатку змінити вхідний текстовий файл і перекомпілювати програму.

Якщо користувач авторизований, тобто вибрав текстовий файл і натиснув на клавішу “Submit” в нього є доступ до навігаційного меню, яке надає можливість швидко здійснювати навігацію по основним функціям системи. Є можливість переглянути всі основні параметри, такі як:

– тиск на вході в компресор, Па;
– молярний потік повітря, моль/с;
– температура повітря, К;
– назва ТЕЦ;
– ККД газотурбінної установки;
– назва і порядковий номер газотурбінної установки.
– зміна масової частки H_2O в газовій суміші;

– зміна масової частки N_2 в газовій суміші;
– зміна масової частки O_2 в газовій суміші;
– зміна масової частки CH_4 в газовій суміші;
– зміна масової частки CO_2 в газовій суміші.

Ця інформація показана на рисунку 4.4.

Рисунок 4.4 — Візуалізація основних значень

У користувача є можливість переглядати як одну частину інформації, так і інформацію із всього файлу. Щоб зробити це, йому необхідно натиснути на клавішу “Показати все”. Якщо ця дія пройшла успішно, буде здійснено перехід до сторінки зі списком всіх ТЕЦ і списком всіх газотурбінних установок а також надано відповідну інформацією про них. Користувач системою може самостійно визначати яку інформацію йому потрібно переглядати і обирати за допомогою відповідних клавіш ту чи іншу функцію системи. Відповідний компонент зображено на рисунку 4.5.

За допомогою допоміжних фільтрів користувач може фільтрувати параметри, які він хоче побачити. Дані функціональні можливості зображено на рисунку 4.6.

Для переходу до компоненту перегляду списку ТЕЦ або газотурбінних установок потрібно натиснути на відповідну кнопку навігаційного меню. Для кожного елементу списку доступна вся інформація по цьому параметру. Ми можемо це побачити на рисунку 4.7.

Для переходу до візуалізації елементів потрібно натиснути на необхідну кнопку списку і буде здійснено автоматичний показ або навпаки, приховування інформації необхідного параметру. Ми можемо це побачити на рисунку 4.8.

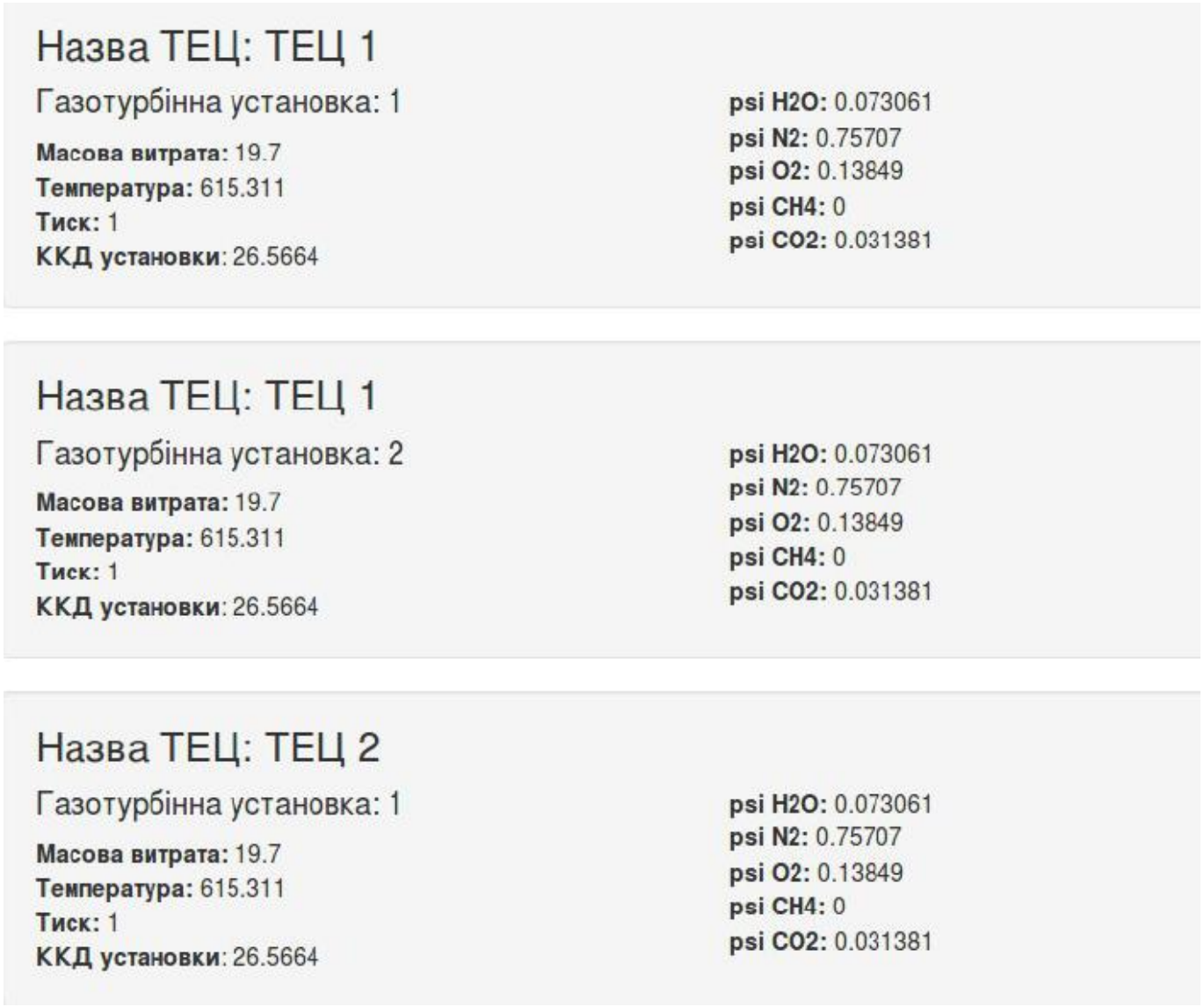

Рисунок 4.5 — Візуалізація результатів роботи декількох ТЕЦ

Якщо дані, що внесені в тестовий файл відправки, не проходять валідацію по кодуванню, то даний текстовий файл буде недоступним для відображення.

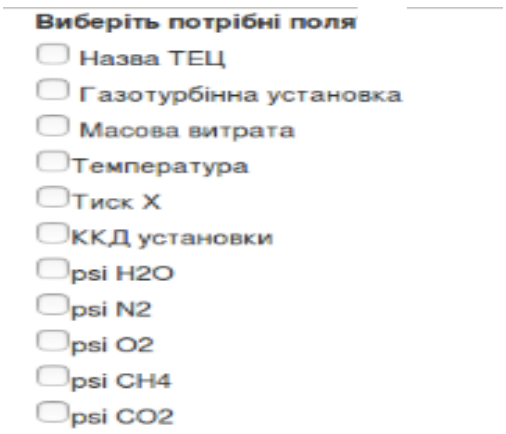

Рисунок 4.6 — Вибір полів для візуалізації

Рисунок 4.7 —Візуалізація параметрів після вибору полів

Система надає користувачу можливість фільтрації даних по символах на всі види параметрів. Для цього створено діалогове вікно, у яке необхідно вписати декілька символів, і після цього система відобразить всі результати, які містять набір цих символів(рисунок 4.8). Якщо ж система не знайде відповідні результати, то сторінка буде залишатись порожньою доти, поки користувач не введе такі символи, які спричинять хоча б співпадіння і результати одразу будуть відображені.

Рисунок 4.8 – Вікно фільтрації полів

Для відображення діаграм створено окреме діалогове вікно. Воно відображає результати роботи у формі стовбчикової діаграми, на якій можна побачити і порівняти значення певних параметрів на різних ТЕЦ в один і той самий момент часу.

Відображення діаграм відбувається з такими параметрами порівняння:

– молярний потік повітря, моль/с;
– температура повітря, К;

– ККД газотурбінної установки;
– зміна масової частки H_2O в газовій суміші;
– зміна масової частки N_2 в газовій суміші;
– зміна масової частки O_2 в газовій суміші;
– зміна масової частки CH_4 в газовій суміші;
– зміна масової частки CO_2 в газовій суміші.

Приклад діаграм можна побачити на рисунках 4.9-4.13

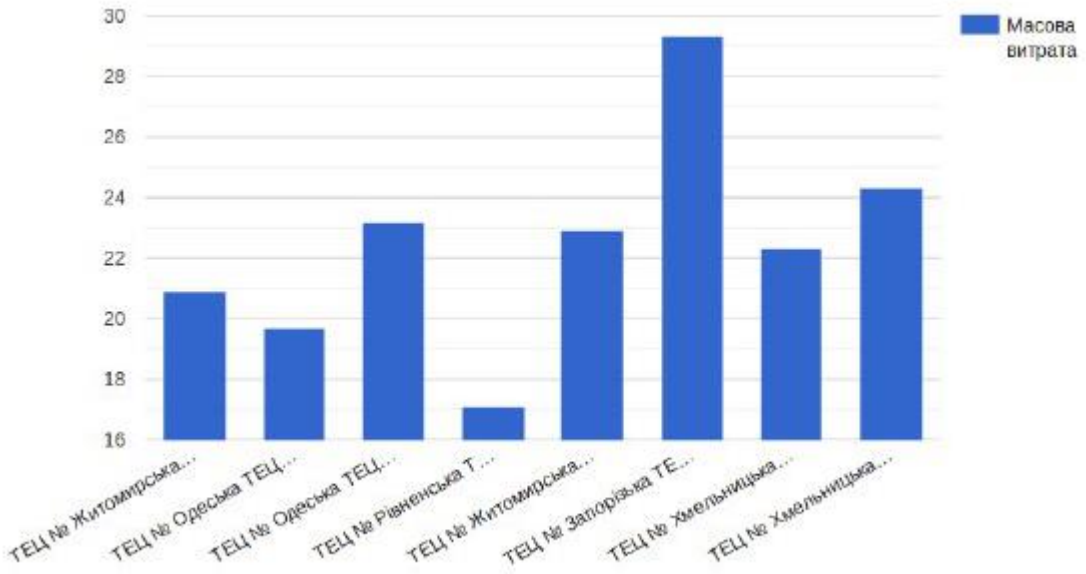

Рисунок 4.9 – Порівняльна діаграма масової витрати палива для всіх ТЕЦ

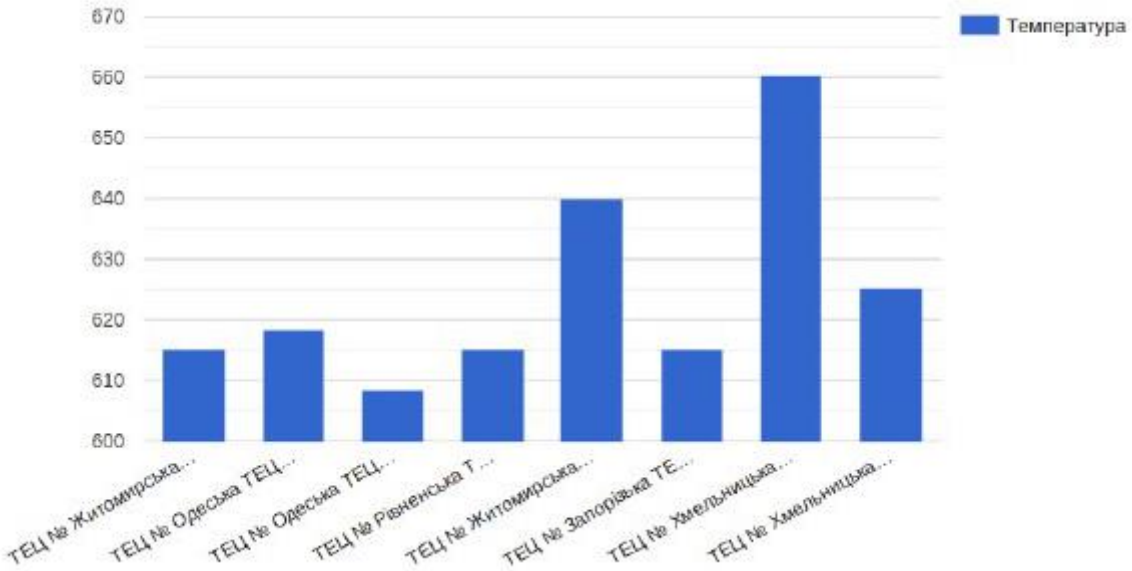

Рисунок 4.10 – Порівняльна діаграма температури для всіх ТЕЦ

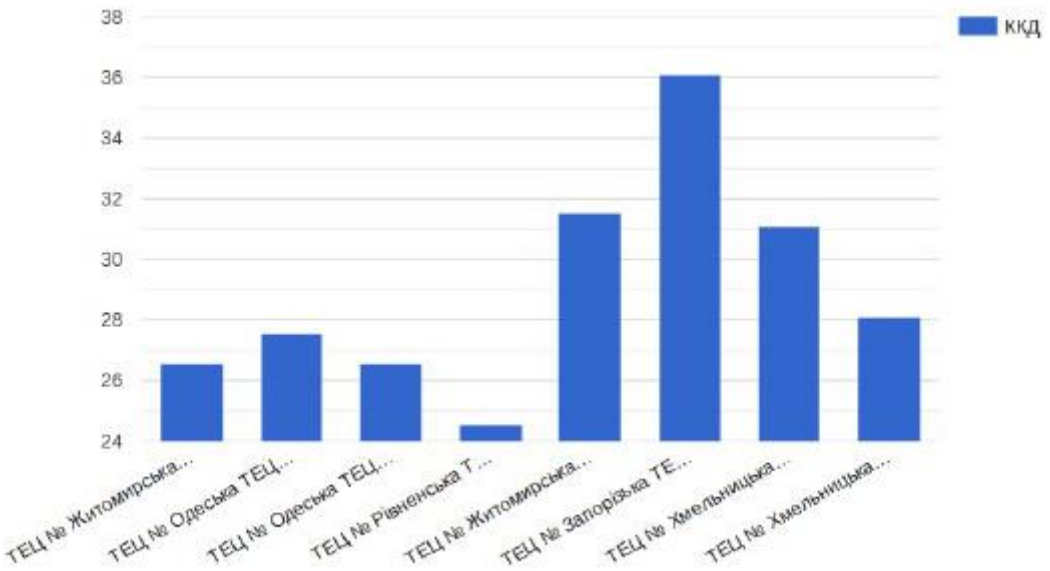

Рисунок 4.11 – Порівняльна діаграма ККД для всіх ТЕЦ

При наведенні курсора на один із стовбчиків показується діалогове вікно, у якому показується інформація про певний параметр цієї ТЕЦ(рисунок 4.12).

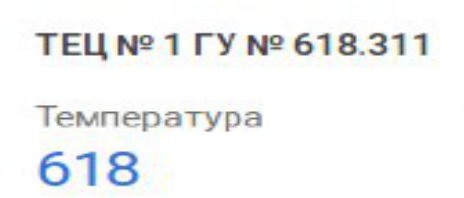

Рисунок 4.12 –Вигляд діалогового вікна при наведенні курсора

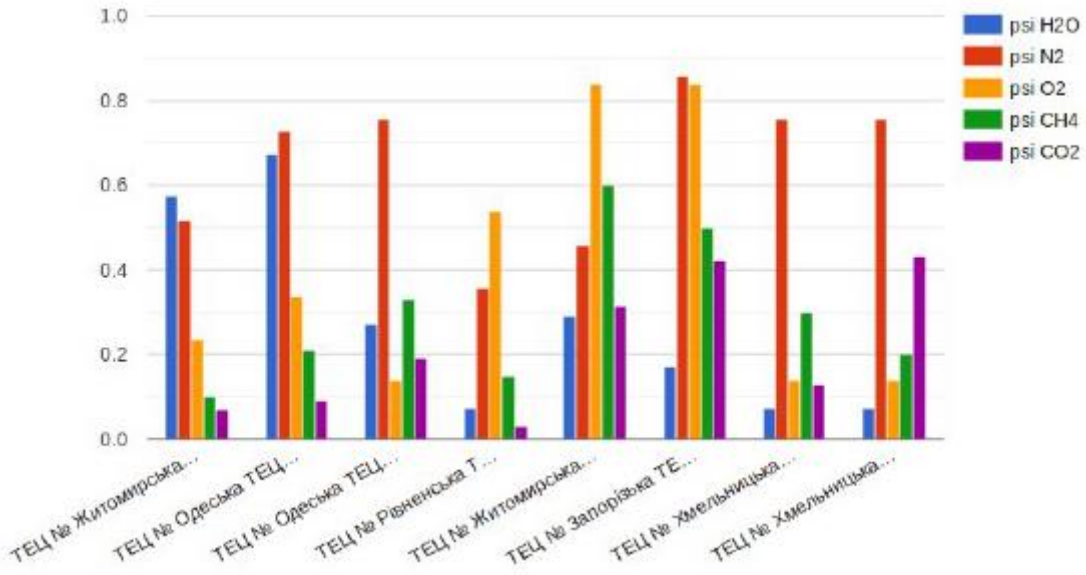

Рисунок 4.13 – Порівняльна діаграма масових часток елементів для всіх ТЕЦ

Google Charts використовує методи класів HeadHelper і DiagramStructure, за допомогою яких зчитує та виводить дані у вигляді діаграм. HeadHelper містить основні параметри за якими створюються діаграми. Клас DiagramStructure місить метод Save(), що зберігає інформацію у такому вигляді.

ВИСНОВКИ

У ході виконання дипломної роботи було створено систему, що містить функціональні можливості для візуалізації результатів моделювання робочого процесу газотурбінної установки за допомогою веб-технологій.

Програмний продукт написано на мові Python та за допомогою фреймворків Flask і jQuery.

Створена система дозволяє виконувати задачі:

– візуалізація результатів робочого процесу газотурбінної установки, яка дозволяє відобразити за допомогою веб-браузера параметри системи в певний момент часу роботи системи за допомогою наданого текстового файлу;

– зчитування текстового файлу з результатами роботи газотурбінної установки;

– порівняння результатів робочого процесу газотурбінної установки, яка дозволяє порівняти результати роботи з використанням декількох газотурбінних установок за допомогою діаграм.

Програмний продукт є універсальним і може бути використаний для більш масштабного проекту, який спеціалізується на моделюванні ГУ та промислових ТЕЦ.

Користувач має змогу власноруч задавати всі параметри вхідних даних у вхідному файлі. На виході генерується веб-сторінка, яка і є результатом виконання системи візуалізації результатів, та представлені результати у вигляді списку та діаграм.

Користувач має змогу змінити конфігурацію сторінки, змінюючи поля і параметри для візуалізації результатів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кириллов И.И. Газовые турбины и газотурбинные установки. Том II. Газотурбинные установки /И.И. Кириллов – М.: Машгиз, 1956. — 318 с.
2. Арсеньев Л.В., Тырышкин В.Г. Газотурбинные установки: Конструкции и расчет. Справочное пособие / Л.В. Арсеньев, В.Г. Тырышкин — Л.: Машиностроение, 1978. — 232 с.
3. Уваров В.В. Газовые турбины и газотурбинные установки. Учебное пособие для машиностроительных вузов / В.В Уваров – М., "Высшая школа", 1970 – 320 с.
4. Цанев С.В., Буров В.Д., Ремезов А.Н. Газотурбинные и парогазовые установки тепловых электростанций. Учебное пособие для вузов. — М.: МЭИ, 2002. — 584 с.
5. Larminie, James. Dicks, Andrew. Fuel Cell Systems Explained, 2nd Edition, Wiley 2003.
6. Fundamentals of Thermodynamics, Richard E. Sonntag, Claus Borgnakke, Gordon J. Van Wylen, 6.Aufl., 2003, John Wiley & Sons.
7. Cengel,Y.A., Thermodynamic: An Engineering approach, 5th ed, McGraw-Hill, 2006.
8. Щегляев А.В. Паровые турбины. Том 1 - Теория теплового процесса и конструкции турбин. Учебник для вузов: 6-е изд., переработанное, дополненное проф. Б. М. Трояновским/ А.В. Щегляев – М.:Энергоатомиздат, 1993 – 384 с.
9. Зрелов В.А. Отечественные газотурбинные двигатели. Основные параметры и конструктивные схемы /В.А. Зрелов – М.: Машиностроение, 2005. — 336 с.
10. Michael Hoffmann — Why I Switched From Visual Studio Code To JetBrains WebStorm — [Електронний ресурс] — 2019 —
11. Зиновьев А. Ю., Визуализация многомерных данных, Красноярск, Изд. КГТУ, 2000.

ДОДАТОК А

Система візуалізації результатів моделювання робочого процесу за допомогою веб-технологій

Специфікація

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5169_19Б

Аркушів 2

Київ – 2019

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ" КПІ ім. Ігоря Сікорського"_ТЕФ_АПЕПС_ТР5169_19Б	Записка.doc	Пояснювальна записка
Компоненти		
УКР.НТУУ" КПІ ім. Ігоря Сікорського"_ТЕФ_АПЕПС_ТР5169_19Б 12-1	controller.cs	Контроллер для обробки запитів
УКР.НТУУ" КПІ ім. Ігоря Сікорського"_ТЕФ_АПЕПС_ТР5169_19Б 12-2	ChartsCreator.cs	Сервіс для створення діаграм
УКР.НТУУ" КПІ ім. Ігоря Сікорського"_ТЕФ_АПЕПС_ТР5169_19Б 12-3	downloadHendler.cs	Базовий сервіс для завантаження вхідного файлу

ДОДАТОК Б

Система візуалізації результатів моделювання робочого процесу за допомогою веб-технологій

Лістинг програмного модуля

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ТР5169_19Б 12-2

Аркушів 7

Київ – 2019

```
google.charts.load('current', {'packages':['corechart', 'bar']});

function createMassFlowChart(data) {
    let res = [];
     for (let key in data){
         for (let el in data[key]){
            console.log(data[key][el], '11111');
            let obj = data[key][el];
            let primary_key = "ТЕЦ № "+ obj.tec + " ГУ № " + obj.gaz;
             res.push([primary_key, +obj.mas_vytrat])
         }
    };
    let column1 = '№ ТЕЦ, № Установки';
    let column2 = 'Масова витрата';
    let element_id = 'chart1';
    let title = 'Динаміка зміни масової витрати газових установок ТЕЦ: ';
    let subtitle = 'в кг/с';

     google.setOnLoadCallback(()=>drawChart(res,column1,column2, element_id,
title, subtitle));
     // drawChart(res)
    }

function createTemperatureChart(data) {
     let res = [];
     for (let key in data){
         for (let el in data[key]){
            let obj = data[key][el];
            let primary_key = "ТЕЦ № "+ obj.tec + " ГУ № " + obj.gaz;
             res.push([primary_key, +obj.temp])
         }
    };
    let column1 = '№ ТЕЦ, № Установки';
    let column2 = 'Температура';
    let element_id = 'chart2';
    let title = 'Динаміка зміни температури газових установок ТЕЦ: ';
    let subtitle = 'в °С';

     google.setOnLoadCallback(()=>drawChart(res,column1,column2, element_id,
title, subtitle));
}

function createKKDChart(data) {
     let res = [];
     for (let key in data){
         for (let el in data[key]){
            let obj = data[key][el];
            let primary_key = "ТЕЦ № "+ obj.tec + " ГУ № " + obj.gaz;
             res.push([primary_key, +obj.kkd_ustanovky])
         }
    };
    let column1 = '№ ТЕЦ, № Установки';
    let column2 = 'ККД';
    let element_id = 'chart3';
    let title = 'Динаміка зміни ККД газових установок ТЕЦ: ';
    let subtitle = 'у % ';
```

```
     google.setOnLoadCallback(()=>drawChart(res,column1,column2, element_id,
title, subtitle));
}


function drawChart(res,column1,column2, element_id, title, subtitle) {

  var data = new google.visualization.DataTable();
  data.addColumn('string', column1);
  data.addColumn('number', column2);

  data.addRows(res);

  var options = {
    chart: {
      title: title,
      subtitle: subtitle
    },
    width: 900,
    height: 500
  };

    var chart = new google.visualization.ColumnChart(
        document.getElementById(element_id));

      chart.draw(data, options);
}


function psiCharts(data) {
     let res = [];
     for (let key in data){
         for (let el in data[key]){
            let obj = data[key][el];
            let primary_key = "ТЕЦ № "+ obj.tec + " ГУ № " + obj.gaz;
             res.push([primary_key, +obj.psi_h20, +obj.psi_n2, +obj.psi_o2,
+obj.psi_ch4, +obj.psi_co2])
         }
    };
     google.setOnLoadCallback(()=>drawPSIChart(res));
}

function drawPSIChart(res,column1,column2, element_id, title, subtitle) {

  var data = new google.visualization.DataTable();
  data.addColumn('string', '№ ТЕЦ, № Установки');
  data.addColumn('number', 'psi H2O');
  data.addColumn('number', 'psi N2');
  data.addColumn('number', 'psi O2');
  data.addColumn('number', 'psi CH4');
  data.addColumn('number', 'psi CO2');

  data.addRows(res);

  var options = {
    chart: {
      title: 'Динаміка зміни різних PSI газових установок ТЕЦ: ',
      subtitle: 'у % '
```

```
    },
    width: 900,
    height: 500
  };

    var chart = new google.visualization.ColumnChart(
        document.getElementById('chart4'));

      chart.draw(data, options);

function ajaxQueryFunction() {
  //   var form = new FormData();
  //
  //   form.append("plantFile",_file.files[0]);
  //   console.log(form);
  //   var xhr = new XMLHttpRequest();
  //   xhr.open('POST', '/file_downloading', true);
  //
  //   xhr.onload = function () {
  // // Запрос завершен. Здесь можно обрабатывать результат.
  //   };

    // function readFile(event) {
    //   textarea.textContent = event.target.result;
    //   console.log(event.target.result);
    // }

    const request = new XMLHttpRequest();
        let _file = document.getElementById("plantFile");
        let file_data = _file.files[0];
        console.log(file_data);

         // var reader = new FileReader();
         // reader.addEventListener('load', readFile);
         // reader.readAsText(_file);

        request.open('POST','/file_downloading', true);
        xmlHttpRequest.setRequestHeader("Content-Type", _file.type);
        console.log(_file.type)

        request.onload = () => {

            const data = JSON.parse(request.responseText);

            if (data.success){
                console.log(data.res);
                // const contents = `1 USD is equal to ${data.rate}
${currency}.`;
                // document.querySelector('#result').innerHTML = contents
            }
            else {
                // document.querySelector('#result').innerHTML = 'There was
an error.';
            }
        };

        // let ss = reader.readAsDataURL(file_data);
        // console.log(ss)
```

```
        const formData = new FormData();
        formData.append('file1', _file);

        //sen request
        request.send(formData);
        return false;
}



$( document ).ready(function(){
            //Perform Ajax request.
            $.ajax({
                url: '/get_data',
                type: 'get',
                success: function(data){
                    //If the success function is execute,
                    //then the Ajax request was successful.
                    //Add the data we received in our Ajax
                    //request to the "content" div.
                    $('#content').html(data);
                    createMassFlowChart(data);
                    createTemperatureChart(data);
                    createKKDChart(data);
                    psiCharts(data);
                    for (let key in data){
                        createTemplate(data[key]);
                    };
                },
                error: function (xhr, ajaxOptions, thrownError) {
                    var errorMsg = 'Ajax request failed: ' +
xhr.responseText;
                    $('#content').html(errorMsg);
                  }
            });
        });
var idCounter = 0;
function createTemplate(data) {
    // console.log('111111',data);


    function reportTemplate(report) {
        console.log(report);
        idCounter++;
        return `
            <div class="row" id=report_id_${idCounter}>
            <div class="col-md-12">
                <div class="well">
                <div class="row">
                    <div class="col-xs-12">
                        <p class="h3 p0">Назва ТЕЦ: ${report.tec}</p>
                    </div>
                    <div class="col-xs-6">
                        <div class="p1"><b><p class="h4">Газотурбінна
установка: ${report.gaz}</p></b></div>
                        <div class="p2"><b>Масова витрата: </b>
${report.mas_vytrat} кг/с<br/></div>
```

```
                        <div class="p3"><b>Температура: </b> ${report.temp}
°C<br/></div>
                        <div class="p4"><b>Тиск: </b> ${report.tysk}
бар<br/></div>

                        <div class="p5"><b>ККД установки</b>: </b>
${report.kkd_ustanovky} %<br/></div>
                    </div>
                    <div class="col-xs-6">
                        <div class="p6"><b>psi H2O: </b>
${report.psi_h20}</div>
                        <div class="p7"><b>psi N2: </b>
${report.psi_n2}</div>
                        <div class="p8"><b>psi O2: </b>
${report.psi_o2}</div>
                        <div class="p9"><b>psi CH4: </b>
${report.psi_ch4}</div>
                        <div class="p10"><b>psi CO2: </b>
${report.psi_co2}</div>
                    </div>

            </div>
            </div>
        </div>
        </div>
            `
    }
    var result = [];
    Object.keys(data).map(function(key) {
        result.push(data[key]);
    });
    // console.log('222222', result);

    document.getElementById("dataResult").innerHTML +=
`${result.map(reportTemplate).join('')}`

}


import re


class FileParser:

    @staticmethod
    def parse():
        with open('powerPlants-4.txt', 'r', encoding='windows-1251') as file:
            _file = file.readlines()

        tec_items = {}

        tec_data = {}

        tec_key = None
        gaz_key = None
        for line in _file:
            if 'Назва ТЕЦ' in line:
                tec_data = {}
```

```
                tec_key = re.findall("\d+(?:\.\d+)?", line)[0]
                tec_name = line.split(":")[1]
                if tec_key not in tec_items:
                    tec_items[tec_key] = {}

            if 'Газотурбінна установка' in line:
                gaz_key = re.findall("\d+(?:\.\d+)?", line)[0]
                if gaz_key not in tec_items[tec_key]:
                    tec_items[tec_key][gaz_key] = {'tec': tec_name, 'gaz':
gaz_key}

            if 'Масова витрата' in line:
                tec_items[tec_key][gaz_key]['mas_vytrat'] =
re.findall("\d+(?:\.\d+)?", line)[0]
            if 'Температура' in line:
                tec_items[tec_key][gaz_key]['temp'] =
re.findall("\d+(?:\.\d+)?", line)[0]
            if 'Тиск' in line:
                tec_items[tec_key][gaz_key]['tysk'] =
re.findall("\d+(?:\.\d+)?", line)[0]
            if 'psi H2O' in line:
                tec_items[tec_key][gaz_key]['psi_h20'] =
re.findall("\d+(?:\.\d+)?", line)[1]
            if 'psi N2' in line:
                tec_items[tec_key][gaz_key]['psi_n2'] =
re.findall("\d+(?:\.\d+)?", line)[1]
            if 'psi O2' in line:
                tec_items[tec_key][gaz_key]['psi_o2'] =
re.findall("\d+(?:\.\d+)?", line)[1]
            if 'psi CH4' in line:
                tec_items[tec_key][gaz_key]['psi_ch4'] =
re.findall("\d+(?:\.\d+)?", line)[1]
            if 'psi CO2' in line:
                tec_items[tec_key][gaz_key]['psi_co2'] =
re.findall("\d+(?:\.\d+)?", line)[1]
            if 'ККД установки' in line:
                tec_items[tec_key][gaz_key]['kkd_ustanovky'] =
re.findall("\d+(?:\.\d+)?", line)[0]

        return tec_items
```

ДОДАТОК В

Система візуалізації результатів моделювання робочого процесу за допомогою веб-технологій

Опис програмного модуля

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ТЕФ_АПЕПС_ ТР5149_19Б 12-2

Аркушів 8

Київ – 2019

ЗМІСТ

-3-

ЗАГАЛЬНІ ВІДОМОСТІ

Програмний код включає в себе частину архітектури системи, що використовується для обробки запиту клієнту.

Запит оброблюється відповідними методами класу Controller, який також гарантує а доступ до даних текстового файлу.

Для отримання / редагування даних метод контроллеру звертається до відповідного сервісу.

Веб-додаток реалізовано як API за допомогою Flask та використовуючи jQuery для задання шаблону звертань до даних.

-4-

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Програмний продукт надає такі функціональні можливості:

— зчитування текстового файлу;
— візуалізація результатів робочого процесу газотурбінної установки;
— вивід порівняльних результатів роботи у вигляді діаграм.

-5-

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Робота з системою здійснюється за принципом “клієнт-сервер”

Запит з клієнта за допомогою системи зчитування опрацьовуються за вказаним URL. Для кожної операції, що виконується користувачем є відповідний метод контроллера. Для того, щоб відкрити новий файл, клієнт надсилає запит POST з відповідною моделлю на URL “http://127.0.0.1:5000/start_page”, після отримання запиту викликається метод Post контроллера. Для обробки файлу викликається метод Add відповідного сервісу, що надає можливість візуалізувати результати роботи на екрані користувача.

-6-

ТЕХНІЧНІ ЗАСОБИ, ЩО ВИКОРИСТОВУВАЛИСЯ

Систему було створено мовою програмування Python у середовищі JetBrains PyCharm 2018 для серверної частини та мовою jQuery та JavaScript для клієнту.

Для створення API було використано Flask, для візуалізації даних використовуються HTML, CSS та Bootstrap.

-7-

ВХІДНІ ТА ВИХІДНІ ДАНІ

Вхідними даними для створення системи візуалізації результатів робочого процесу газотурбінної установки є текстовий файл, який містить наступні дані:

– тиск на вході в компресор, Па;
– молярний потік повітря, моль/с;
– температура повітря, К;
– тиск повітря, Па;
– назва ТЕЦ;
– ККД газотурбінної установки;
– назва і порядковий номер газотурбінної установки.

До вихідних даних відносяться:

– зміна масової витрати газу для кожного із трьох станів в залежності від часу;
– зміна температури газу для кожного із трьох станів в залежності від часу;
– зміна тиску газу для кожного із трьох станів в залежності від часу;
– зміна масової частки H_2O в газовій суміші для кожного із трьох станів в залежності від часу;
– зміна масової частки N_2 в газовій суміші для кожного із трьох станів в залежності від часу;
– зміна масової частки O_2 в газовій суміші для кожного із трьох станів в залежності від часу;
– зміна масової частки CH_4 в газовій суміші для кожного із трьох станів в залежності від часу;
– зміна масової частки CO_2 в газовій суміші для кожного із трьох станів в залежності від часу.